Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних інформаційних систем

Алгоритми та складність

Лабораторна робота 1

Виконав студент 2-го курсу

групи К-29

Прокопчук Роман

2022

**Завдання**

Реалізувати ідеальне хешування для вектора комплексних чисел

**Теорія**

Ідеальне хешування – спосіб хешування, який перетворює статичну множину ключів у комірки хеш-таблиці без колізій.

**Алгоритм**

1)Хешуємо елементи множини в комірки з підтаблицями за допомогою хеш-функції для вектора, яка має наступний вигляд :

*, i= –* значення універсальної хеш-функції для *i-го* елемента вектора , яка має вигляд ((, де -ключ, , – параметри, просте число, підібране так, щоб , ; Для комплексного числа ключ обчиcлюємо так :

де - відносно велике просте число, – дійсна та умовна частини комплексного числа

просте число

кількість підтаблиць.

2)Виділяємо пам’ять для кожної підтаблиці так, щоб їх розмір дорівнював квадрату кількості ключів, захешованих в підтаблицю. Також, для кожної таблиці задаємо параметри для універсальної функції, щоб здійснити вторинне хешування за допомогою хеш-функції для вектора

3)Хешуємо елементи вторинною хеш-функцією та записуємо в комірки елементи множини

**Складність**

Побудова : де -кількість елементів у множині, розмір вектора

Пошук :

**Мова програмування**

C++

**Модулі програми**

* Файли data\_types.h/data\_types.cpp – містять опис класу ComplexNumber:

конструктори, «геттери» та оператори == i <<

* lab\_1.cpp

1. UniversalHash – структура, за допомогою якої ми виконуємо хешування комплексних чисел(універсальне хешування)
2. VectorHash - структура, за допомогою якої ми виконуємо хешування вектора комплексних чисел
3. HashSet – клас, у якому реалізовується ідеальне хешування. Він містить конструктор, методи Has для перевірки належності певного числа до хеш-таблиці, Print для друку вмісту; допоміжну структуру Bucket для представлення підтаблиці та параметрів для хешування, та допоміжні методи для визначення позицій в хеш-таблиці
4. Assert-допоміжна функція для перевірки роботи методу Has

**Інтерфейс користувача**

Дані виводяться в консоль

**Тестові приклади**

Input:

Кількість підтаблиць : 3

Універсальна хеш-функція : (

Для хешування векторів виберемо ;

Покажемо, як працює ідеальне хешування на першому векторі :

*h(*

*h(*

Значення хеш-функції для вектора :

Прохешувавши первинною хеш-функцією вектори, отримаємо, що в нульову підтаблицю попадають 0 векторів, першу – 2, другу - 1

Далі для вторинного хешування першого вектора виберемо наступні параметри для другої підтаблиці:

*h(*

*h(*

*mod =0*

Отже, вектор 1 поміститься в нульову комірку другої підтаблиці.

Якщо виконати всі відповідні дії і обрати відповідні параметри, отримаємо таку хеш таблицю

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | 0 | | 1 | | 2 | | 3 | |
| 0 | | | | | |  | |  | |  | |  | |
| 1 | | | | | | 2 | |  | |  | | 3 | |  | |  | |  | |  |  |
| 2 | | | | | | 1 | |  | |  | |  | |  | |  | |  | |  |  |
|  |  |  |  |  |  | |  | |  | |  | |  | |  | |  | |
|  |  |  |  |  |  | |  | |  | |  | |  | |  | |  | |

**Висновки**

Ідеальне хешування – ефективний спосіб зберігання статичної множини елементів, який дозволяє дуже швидко шукати елементи в хеш-таблиці.

**Література**

* Лекція 1